# Starlink cheat sheet: Overview

**Most useful packages for JCMT**: **KAPPA** (various analysis tools), **SMURF** (gridding/DR), **CUPID** (source/clump finding), **CONVERT** (convert files), **GAIA** (GUI based visualisation and analysis, including data cubes), **TOPCAT** (catalogue GUI), **SPLAT** (spectra GUI), **stilts** (catalogue commandline). **ORAC-DR**: Data reduction pipelines. **PICARD**: data analysis pipelines.

## Starlink data files: `.sdf`

### NDF data format (SUN/33: Overview of an NDF)

- N-Dimensional Hierarchical Data Format: extension `.sdf`
- Go to/from FITS with **CONVERT**'s `ndf2fits`/`fits2ndf`
- Stores coordinate information differently from FITS.
- Each .sdf file can contain multiple NDF structures.
- Can have multiple **coordinate frames** defined.

### Examining NDF files

- `hdstrace` list contents of a file (data and metadata).
- `ndftrace` shows the coordinate information.
- `fitslist` shows the FITS header (not coordinate info).
- `stats` Calculate statistics on a file/subset.
- use **GAIA** to view any components/extension arrays.

### Useful features of JCMT NDFs

- **History tracking:** View with **KAPPA**'s `hislist`.
- **Variance & Error Components:** in gridded data, access/view with `comp=var` or `comp=err`.
- **Exposure time map:** in gridded data, access as `mysdf.more.SMURF.exp_time`.
- **Quality Arrays** can indicate extra information about map, e.g. makemap masks.
- **Provenance Tracking**: view with `provshow`.
- Missing/invalid data is indicated by **`bad`** pixel values (like NaN) and handled sensibly by all commands.

## Tips for running Starlink commands

- Starlink commands take **positional and/or keyword parameters.**
- **Commands will prompt user** for any missing *required* parameter values.
- **Type `?`** at prompt to see more information on that parameter.
- **Optional parameters** will use a default value if not given to the command.
- Often, optional parameters will use values from your **last run as the default.**
- Global defaults, last used & output values stored in **ADAM directory: (~/adam/)**
- Files in the **ADAM directory** can become corrupted; delete if needed.
- Turn off run-time defaults by **passing `RESET`** to the command.
- **Include `PROMPT`** on command line to force prompting of every parameter.
- **Give `ACCEPT`** to accept defaults of all unlisted parameters without prompting.
- Control amount of output to screen with: **`msg_filter=verbose` or `quiet`**
- To give a NULL value to a parameter **type: `!`**
- To exit cleanly from interactive prompts **type: `!!`**
- When specifying parameters on the shell, you may need to **escape with quotes/backslashes any shell special characters;** e.g. `* "',([`
- **Use a caret (^)** before a filename to tell Starlink software to read the contents of that file (e.g. to give list of input files).
- Specify **multiple NDFs**: `in="in1,in2"` or with wildcards: `in="?2010*"`.
- Get **output values** with: `parget <parameter> <commandname>`.

### Environmental Variables

- `ADAM_USER`: Specify ADAM directory. (Don't have multiple invocations running with same ADAM_USER).
- `NOPROMPT`=1: don't prompt for input.

### Scripting Advice

- **Shell scripts**: C-shell cookbook (SC/4).
- **Python**: starlink-wrapper package.
- **Perl**: module Starlink::AMS::Task.

### Getting help

- JCMT Heterodyne cookbook: SC/20
- JCMT SCUBA-2 cookbook: SC/21
- interactive commandline help for packages: type `kaphelp`, `smurfhelp` etc.
- `findme`/`showme` to show html docs.
- HTML and PDF Starlink User Notes (SUNs) available for each package.

## Accessing subregions of a map or cube

### NDF Sections (SUN/95)

- Pass rectangular/cuboid subsections to commands.
- Use integers to index by NDF pixel coordinates: `map.sdf\(-5:5,10:25\)` (rectangle of 11x16 pixels)
- Use floats to index by current astronomy coordinate frame: `map(1h34m10.1s:1h34m12.4s, -2d35m:-2d30m)`
- Can also use centres and extents: `map.sdf\(5~25,-5~25\)` (square of 25x25 pixels)
- Note: you must escape brackets if running in the shell.

### ARD regions (SUN/183)

- Used to extract more complex shapes from cubes/maps.
- Supports rectangles, circles, ellipses, polygons & more.
- Textfile with shapes and positions, in pixel or WCS coords.
- Generate interactively in **GAIA** or **KAPPA**'s `ardgen`
- Subset of ARD can be used in **GAIA**'s imaging toolbox.
- Mask a region of an image using **KAPPA**'s `ardmask` command

Starlink also supports AST regions (see **KAPPA**'s `regionmask`) and IVOA's STC-S regions (plottable within **GAIA**).

## Coordinate systems

- NDF data arrays always have **pixel coordinate system** defined – often with centre of map on 0,0 **(pixel origin)**.
- See NDF pixel coordinates interactively in GAIA by selecting **Image Analysis->Change Coordinates->Show-all coordinates**
- Multiple coordinate frames can be defined; switch between them with `wcsframe` or alter current with `wcsattrib`.
- To regrid, use `regrid` or to align with an existing map use `wcsalign`.

# Starlink cheat sheet: Example commands with usage

**BASH: initalise Starlink via:**

```
export STARLINK_DIR=/path/to/starlink
source $STARLINK_DIR/etc/profile
```

**Packages**: Initialise packages by typing their name.

**TCSH: initalise Starlink via:**

```
setenv STARLINK_DIR /path/to/starlink
source $STARLINk_DIR/etc/cshrc
source $STARLINK_DIR/etc/login
```

## Useful features of the Starlink/Starjava GUIs.

**GAIA** can open and view 2-D maps and 3-D cubes, extract spectra, rebin/average/collapse cubes, make movies, volume and iso-surface render cubes, perform astrometry, perform source finding, plot catalogs, crop images, get statistics, create histograms, perform photometry, search and display VO images and catalogs and much more...
**TOPCAT** is a GUI useful for catalog reading/writing, matching of multiple catalogs, analysis and visualisation. You can also use it to carry out TAP queries, e.g. to query the CADC JCMT Science archive.
**SPLAT(Spectral Analysis Tool)** can compare multiple spectra, identify lines, fit and manipulate spectra and produce plots.

## CONVERT: see SUN/55 for all commands

**Convert on the fly** Launch `convert` to use input/output files in other formats like FITS, through auto conversion.
**Export/import from NDF** supports: FITS, ASCII, IRAF, TIFF, GIF. Commands named like: `ndf2fits` & `fits2ndf`.

## KAPPA: see SUN/95 for all commands

**Display maps/2D NDF sections** `display myndf.sdf\(0:50,100:200,0\) mode=faint device=xw`
**Display spectra/1D NDF section** `linplot myspectra.sdf device=xw` **See SUN/95 §11 for more complex images.**
**Calculate statistics** `stats in=mycube.sdf order=True percentiles=\[0.99,0.95,0.9\]`
**Create SNR map** `makesnr in=mymap.sdf out=mymap_snr.sdf minvar=0.0`
**Clip map at 3$\sigma$ based on variance array** `errclip in=file.sdf out=file_clipped.sdf mode=SNR limit=3`
**Integrate a cube** `collapse in=cube out=integ axis=vrad low=-5.0 high=10.0 estimator=integ`
**Calculate 2nd moment of cube** `collapse in=cube axis=3 low=-100.0 high=50.0 estimator=Iwd`
**Get 1-D profile through map** `profile in=map out=prof start='"15:32 23:40:08"' end='"15:31 23:48:08"'`
**Extract a spectrum** `pluck omc1 pos="5:35,-5:22" axes=3 method=sincgauss params=\[3,5\] out=spec`
**Clip data above/below thresholds** `thresh in=map out=map_thr thrlo=-0.1 thrhi=500.0 newlo=bad newhi=bad`
**Mosaic/Coadd multiple observations** `wcsmosaic in='m51*' out=mos variance=true method=nearest wlim=0.0`
**Align maps/cubes onto same WCS grid** `wcsalign in=cube1 out=cube1_al ref=cube2 method=nearest`
**Regrid cube's velocity axis to different resolution** `sqorst cube out=resol pixscale=1.0 axis=3 mode=pixelscale`
**Smooth a map/cube with a Gaussian** `gausmooth in=file out=file_smooth fwhm=\(5.3,7.8\) orient=30.0`
**Fit a beam to a map and report** `beamfit in=mars.sdf mode=interface beams=1 pos='"0,0"'`
**Copy NDF/section based on template** `ndfcopy mymap out=res like=refmap.sdf likewcs=True trimbad=True`
**Remove wavelength axis from a SCUBA-2 map** `ndfcopy in=3dmap trim=yes out=2dmap`
**Copy bad pixel mask from reference** `copybad mymap.sdf ref=mymask.sdf out=masked.sdf`
**Grow a map into a cube by repeating the data** `manic in=map out=cube axes=\[1,2,0\] ubound=25 lbound=-25`
**Various NDF/scalar maths commands** `maths`, `add`, `sub div`, `mult`, `convolve`, `cdiv`, `cadd`, `csub`, `cmult`
**Change from vel. to freq** `wcsattrib mycube mode=set name=system newval=freq`
**Change from frequency to vrad coords** `wcsattrib mycube mode=set name=system newval=vrad`
**Change to Galactic coordinates (not regridding)** `wcsattrib myfile.sdf mode=set name=system newval=galactic`
**Replace all NaN/Bad values with 0.0** `nomagic in=myfile out=myfile_nobad repval=0.0`
**Change data values values in a section** `chpix in=myfile out=myfile_changed section='"3:5,4"' newval=50.0`

## SMURF: see SUN/258 for all commands

**Grid raw SCUBA-2 data** `makemap in=^raw.lis out=map config=^dimmconfig.lis pixsize=4.0 trim=true`
**Grid raw ACSIS data** `makecube in=^raw.lis out=cube autogrid=True spread=Gauss params='"0,2.5"'`
**Fit spectra** `fit1d in=cube out=fitprofiles config=^fit1dconfig.lis`

## CUPID: see SUN/255 for all commands

**Clumpfinding** `findclumps mymap out=clumps method=fellwalker outcat=clump.FIT config=^fellconf.lis`
**Get data within clump boundaries** `extractclumps mask=clumps data=other out=extract outcat=cat.FIT`
**Find the large scale structure** `findback in=myndf out=background box=\[5,5,1\] rms=!`

## Pipelines: ORAC-DR & PICARD

**Initialise & run ORAC-DR pipeline** `oracdr_<instrumentname> -cwd`
`oracdr -log sf -loop file -files=files.lis -recpars=^myrecpars RECIPE_NAME`
**Run Picard** `picard -log sf -recpars=^myrecpars RECIPE_NAME 'cat files.lis'`